AMENDMENTS TO THE SPECIFICATION

Please replace the paragraphs at page 1, lines 5 to 20, with the following amended paragraphs:

The present application is related to the following commonly owned U.S. patent applications:

U.S. Patent Application No. 10/640,626 entitled "INSTRUMENTING JAVA CODE BY MODIFYING BYTECODES," filed concurrently herewith under Attorney Docket No. 10017135-1;

U.S. Patent Application No. 10/640,620 entitled "USING INTERCEPTORS AND OUT-OF-BAND DATA TO MONITOR THE PERFORMANCE OF JAVA 2 ENTERPRISE EDITION (J2EE) APPLICATIONS," filed concurrently herewith under Attorney Docket No. 10017134-1;

U.S. Patent Application No. 10/640,625 entitled "PROPAGATING WEB TRANSACTION CONTEXT INTO COMMON OBJECT MODEL (COM) BUSINESS LOGIC COMPONENTS," filed concurrently herewith under Attorney Docket No. 10017133-1; and

U.S. Patent Application No. 10/640,619 entitled "USE OF THREAD-LOCAL STORAGE TO PROPAGATE APPLICATION CONTEXT IN JAVA 2 ENTERPRISE EDITION (J2EE) APPLICATIONS," filed concurrently herewith under Attorney Docket No. 200311221-1.

Please replace the paragraph at page 2, lines 11-17, with the following amended paragraph:

In concert with the increased use of eBusiness applications, software architectures for developing them have become more efficient, albeit at the cost of increased architectural complexity. Such architectures typically rely on multiple, dynamically instantiated, and distributed software components to provide highly scalable eBusiness applications. The associated architectural complexity, however, can present significant challenges in developing methods and systems for monitoring transactions performed by the applications.

Please replace the paragraph at page 2, lines 23-29, with the following amended paragraph:

In one aspect of the present invention, a method of monitoring response time of a method or a function associated with a ~~Java~~ JAVA software component is disclosed. The monitoring method calls for inserting an instrumentation code in a bytecode representation of said method or function to effect generation of a start time marker upon start of execution of said method or function and a stop time marker upon completion of execution of said method or function. The start and stop time markers are then utilized to determine a response time of said method or function.

Please replace the paragraph at page 5, lines 27-28, with the following amended paragraph:

FIG. 14 schematically illustrates an Application Response Measurement (ARM) interface in communication with an application and management agents.

Please replace the paragraph at page 6, lines 15-16, with the following amended paragraph:

FIG. 22 schematically depicts an exemplary Common Object Model (COM) object implementing three interfaces.

Please replace the paragraph at page 6, lines 13-27, with the following amended paragraph:

FIG. 1 schematically depicts an exemplary distributed multi-tier Web application architecture 10 in which transaction monitoring agents according to the teachings of the invention are incorporated. The illustrated multi-tier architecture 10 employs a client 12 as an interface for receiving requests from a user. The client 12 can be, for example, a web browser, or alternatively a probe that periodically transmits requests to a web server 14 for testing operations of the system. Without any loss of generality, the client 12 is assumed to be a web browser in the following discussion. The web browser 12 can be running, for example, on a user's desktop, or alternatively, on a PDA or any other suitable platform. The web browser 12 transmits a user's request to a web server 14 that in turn can communicate with an

application server 16 that hosts a number of software components, e.g., Java Server Pages (JSPs) or servlets. The exemplary application server 16 also hosts a transaction monitoring agent 18 according to the teachings of the invention that can monitor performance of selected methods in software components invoked on the application server 16 in response to requests received from the web server 14, as discussed in more detail below. The term "transaction," as used herein, refers generally to a method or a function within a software component.

Please replace the paragraph at page 6, line 28 to page 7 line 5, with the following amended paragraph:

A software component, e.g., a servlet, invoked on the application server 16 may require the services of another application server for performing business logic needed for servicing the request received from the user. In this exemplary embodiment, software components running on the application server 16 can communicate with other software components running on another application server 20. For example, in embodiments in which the application server 16 is a J2EE server, a servlet or a Java Server Page (JSP) running ~~JPS running~~ on the application server 16 can invoke an Enterprise Java Bean (EJB) software component hosted on the application server 20 for performing a desired business logic. For example, if the user is utilizing the web server for on-line shopping, the EJB may keep track of items in the user's shopping cart.

Please replace the paragraph at page 11, lines 14-24, with the following amended paragraph:

Advantageously, different plug-in instruments can be used at different times without re-instrumenting the classes. In other words, during subsequent executions of the class, a different set of instruments 27A-B can be used. Class/method instrumentation (i.e. inserting hooks) is, therefore, decoupled from the behavior of the plugged-in instruments (i.e. execution-time behaviors). The instruments 27A-B can perform various functions or exhibit various behaviors, such as collecting data and monitoring performance of the instrumented methods and classes, debugging ~~debuging~~ code, providing security or anything that requires information about method calls in the instrumented classes. In addition, the classes can be executed with a "null instrument", which simply returns. In addition, each instrumented class

can be bound to a different implementation of the execCallback interface 36, i.e. to a different plug-in instrument 27A-B.

Please replace the paragraph at page 24, lines 12-23, with the following amended paragraph:

A Java virtual machine (JVM) ~~machine (VJM)~~ includes a class loader, and an application server invokes this class loader to load one or more classes. Most application servers, such as the WebLogic Platform from BEA Systems, Inc. San Jose, Calif., include a class load hook, which can be used to execute a user-specified class before the JVM class loader loads a normal class. Depending on the application server, one might have to specify a name of the user-specified class and/or set a property when starting the application server to notify the application server to execute a user-specified class when loading normal classes. The application server might define an interface, and the user-specified class might implement this interface to communicate with the application server. Thus, the user-specified class can be provided with an opportunity to read and modify a class file as it is being given to the class loader. By this mechanism, the BIC instrumentation tool 26 can modify classes as they are being loaded.

Please replace the paragraph at page 45, line 25 to page 46, line 5, with the following amended paragraph:

The present application is related to the following commonly owned U.S. Patent Applications, all of which are hereby incorporated by reference herein in their entirety: U.S. Patent Application No. 10/640,626 entitled "INSTRUMENTING JAVA CODE BY ~~CODE BU~~ MODIFYING BYTECODES," ~~filed concurrently herewith under Attorney Docket No. 10017135-1~~; U.S. Patent Application 10/640,620 entitled "USING INTERCEPTORS AND OUT-OF-BAND DATA TO MONITOR THE PERFORMANCE OF JAVA 2 ENTERPRISE EDITION (J2EE) APPLICATIONS," ~~filed concurrently herewith under Attorney Docket No. 10017134-1~~; U.S. Patent Application No. 10/640,625 entitled "PROPAGATING WEB TRANSACTION CONTEXT INTO COMMON OBJECT MODEL (COM) BUSINESS LOGIC COMPONENTS," ~~filed concurrently herewith under Attorney Docket No. 10017133-1~~; and U.S. Patent Application No. 10/640,619 entitled "USE OF THREAD-

LOCAL STORAGE TO PROPAGATE APPLICATION CONTEXT IN JAVA 2 ENTERPRISE EDITION (J2EE) APPLICATIONS," ~~filed concurrently herewith under Attorney Docket No. 200311221-1~~.